A Framework for Monitoring Agent-Based Normative Systems

Sanjay Modgil
Noura Faci
King's College London
Dept of Computer Science
London, United Kingdom
sanjay.modgil@kcl.ac.uk

Felipe Meneguzzi
Nir Oren
King's College London
Dept of Computer Science
London, United Kingdom
felipe.meneguzzi@kcl.ac.uk

Simon Miles
Michael Luck
King's College London
Dept of Computer Science
London, United Kingdom
simon.miles@kcl.ac.uk

ABSTRACT

The behaviours of autonomous agents may deviate from those deemed to be for the good of the societal systems of which they are a part. Norms have therefore been proposed as a means to regulate agent behaviours in open and dynamic systems, where these norms specify the obliged, permitted and prohibited behaviours of agents. Regulation can effectively be achieved through use of enforcement mechanisms that result in a net loss of utility for an agent in cases where the agent's behaviour fails to comply with the norms. Recognition of compliance is thus crucial for achieving regulation. In this paper we propose a generic architecture for observation of agent behaviours, and recognition of these behaviours as constituting, or *counting as*, compliance or violation. The architecture deploys monitors that receive inputs from observers, and processes these inputs together with transition network representations of individual norms. In this way, monitors determine the fulfillment or violation status of norms. The paper also describes a proof of concept implementation and deployment of monitors in electronic contracting environments.

Categories and Subject Descriptors

D.2.10 [**Software**]: Software Engineering; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multi-agent systems*

Keywords

Monitoring, norms, electronic contracts

1. INTRODUCTION

Recent years have witnessed a growing interest in the use of norms to regulate and coordinate agent behaviours, and so achieve the overall objectives of multi-agent systems. Two approaches have been taken. In the *regimentation* approach [9], adopted for example by electronic institutions [3], agent behaviours are constrained to those specified by norms. Hence, agent autonomy is drastically curtailed, and such regimented systems are less flexible in that only appropriately specified agents can join. In contrast, the *enforcement* approach [1, 2, 6, 14, 18] allows for autonomous agents, and hence the possibility of violation of norms by agents. Enforcement mechanisms are thus required to motivate agent compliance by threatening some loss of utility for agents in the case of violation. The enforcement approach thus requires that agent actions are *monitored*; that is, they must be observable and recognised as complying with or violating norms, in order that the enforcement mechanisms be appropriately applied.

In this paper we describe a generic framework for monitoring of agent behaviours in normative multi-agent systems. We motivate and describe how our approach builds on the *overhearing* approaches to monitoring (e.g., [10]), whereby messages exchanged among agents are observed, and behaviours are inferred from these messages. This contrasts with *intrusive* approaches [8, 15, 20], which assume that the mental states of agents are available for inspection, thus making the design of agent-based systems more complex, and relying heavily on the compliance of agents to communicate the required data.

Our monitoring architecture accounts for the fact that deployment of agents in normative systems requires some assurance that enforcement mechanisms, such as punishments or sanctions, will be employed only as and when appropriate. Any such assurance will partly rely on some measure of certainty that a norm *is reported* as violated if and only if it has *in actuality* been violated. Two features of our approach to monitoring ensure, to the degree that it is possible, that such assurance can be provided.

1. Observers of agent behaviours are explicitly entrusted by the system's participating agents to accurately report on these behaviours.
2. Trusted observers can be any environmental artifact, that may not only observe for messages exchanged, but also more generally report on whether some state of interest holds or not.

This paper also describes how individual norms — *obligations*, *prohibitions*, and *permissions* — can be represented as *Augmented Transition Networks* (*ATN*s) [21] that are processed by monitor agents, together with observations relayed to the monitors by trusted observers, in order to determine the fulfilment and violation status of norms. The key features of the *ATN* representation and processing are as follows.

1. An abstract, general model of norms is assumed.
2. Representation of complex behaviours and states of interest enacted and brought about jointly by groups of agents.
3. Only behaviours specified by the norms are represented; hence, a given *ATN* can represent the same norm specified in any one of a number of multi-agent systems.
4. *ATN* representations of norms are independent of each other, allowing run time addition and removal of norms.

Cite as: A Framework for Monitoring Agent-Based Normative Systems, S.Modgil, N. Faci, F. Meneguzzi, N. Oren, S. Miles and M.Luck, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

Our approach is thus generic and applicable to a range of dynamic open normative systems, including normative *organisations* developed by the kinds of dedicated languages described in [2], as well as electronic contracting frameworks [18] in which contract clauses specify norms that the contract parties are beholden to.

In summary, this paper makes the following contributions to research on monitoring of norm-governed agent behaviours:

1. We propose a *trusted observer* model with observations of agent messages and states of interest, to provide some measure of assurance that enforcement mechanisms are appropriately applied, so encouraging deployment of agents in normative systems.

2. Norms are individually represented as independent ATNs that provide for monitoring of: complex, jointly realised agent behaviours, and; states of interest brought about by such behaviours.

3. Together, the above provide a general framework that is applicable to monitoring of norms in a broad range of open, dynamic multi-agent systems.

This paper is organised as follows. Section 2 describes some general normative concepts that our approach to monitoring makes use of. Section 3 then motivates and describes an architectural overview of our approach. In Section 4, we describe how individual norms are represented as *ATN*s, and processed by monitor agents. Section 5 describes validation of our approach. We report on a proof of concept implementation of a monitoring agent, and its processing of *ATN* representations of norms encoded in an electronic contract specified by the CONTRACT project[1]. The implementation demonstrates monitoring of *AgentSpeak(L)* agents [19] whose interactions are governed by normative clauses specified in an aerospace contract. Finally, Section 6 concludes with a discussion of future and related work. In particular we discuss how the framework presented here extends and generalises a preliminary framework described in [4].

2. A GENERAL MODEL OF NORMS

In this section we review a recent general model of norms [18] that distinguishes some general normative concepts shared by some existing work on norms and normative systems [12, 5], and which we adopt in this paper. In [18], a norm $\mathcal{N}$ is modelled as a tuple:

$(NormType, NormActivation, NormCondition, NormExpiration, NormTarget)$

where:
$\mathcal{N}$ is said to come into force, or is *activated*, if the conditions, or *state of interest*, described by $NormActivation$ hold. It is when $\mathcal{N}$ is activated, that one must monitor to ensure that the goal, or state of interest, described by $NormCondition$:

- must be brought about by $\mathcal{N}$'s $NormTarget$ in the case that $\mathcal{N}$'s $NormType$ is obligation;
- may be brought about by $\mathcal{N}$'s $NormTarget$ in the case that $\mathcal{N}$'s $NormType$ is permission; or
- must not be brought about by $\mathcal{N}$'s $NormTarget$ in the case that $\mathcal{N}$'s $NormType$ is prohibition.

[1]www.ist-contract.org

Such states of interest describe states of the world in which actions have been performed (e.g., messages sent) or certain properties hold (e.g., 'the temperature is maintained above 23 degrees for at least 90% of the time'). Finally $\mathcal{N}$'s $NormExpiration$ denotes the state of interest under which the norm is no longer in force. Henceforth, we will refer to a norm's $NormActivation$, $NormCondition$ and $NormExpiration$ as a norm's *components*.

Note that in this paper we monitor the class of obligations whose violation can be determined with respect to some temporal condition holding. Without a temporal condition on the satisfaction of an obligation to realise or achieve a state of affairs (an *achievement* obligation), it would not be possible to determine a point in time at which a violation had occurred. Obligations to maintain certain states of affairs (*maintenance* obligations) often also explicitly reference temporal conditions that can be evaluated to determine violation (consider the above mentioned obligation to maintain the temperature for a certain period of time). However, some maintenance obligations do not (e.g., an obligation to always drive on the left), and these will be considered further in future work in Section 6.

EXAMPLE 1. *Consider the norm — $NormGoods$ — that describes an obligation on the purchaser of goods G from a supplier S, where the purchaser is an organisational entity consisting of two agents: the buyer B and the financial department F:*

- $NormType$ = obligation
- $NormActivation$ = B is notified by S that goods G are in stock
- $NormCondition$ =
  - *1)* B must cancel the order within 7 days of receipt of notification
    or
  - *2)* B must accept the order within 7 days of receipt of notification ***and*** F must deposit payment for G in S's bank account, within 3 days of B's acceptance.
- $NormExpiration$ =
  - *1)* B has canceled the order within 7 days of receipt of notification
    or
  - *2)* B has accepted within 7 days of receipt of notification and S has received payment for G from F within 3 days of B's acceptance
    or
  - *3)* S is declared bankrupt
- $NormTarget$ = B, F

Recall that $NormExpiration$ denotes the state of interest under which a norm is no longer in force. The above example illustrates that a norm may be deemed to have expired for reasons other than that the norm's conditions have been fulfilled. In principle, extra conditions of the type 'S is declared bankrupt' may be encoded as exceptions in the activation condition. However, $NormGoods$ may then inappropriately remain in force; suppose that S goes bankrupt three days after B is notified by S that goods G are in stock. The norm will remain in force if 'S is declared bankrupt' is not encoded in the expiration condition.

Finally, note that a number of options obtain as to how one formally represents the states of interest described by a norm's components. In the following section we discuss how these options are determined by issues such as ease of recognition and trust, and how the choice of option may determine the readiness with which agents are deployed in normative systems.

3. TRUSTED OBSERVERS AND THE MONITORING ARCHITECTURE

3.1 Motivating Trusted Observers

Enforcement mechanisms are required to motivate agent compliance with norms. Hence, agent actions must be monitored; that is, they must be observed and recognised as complying with or violating norms, in order that the enforcement mechanisms can be appropriately applied.

Monitoring of compliance requires detecting whether the states of interest described by $NormActivation$, $NormCondition$ and $NormExpiration$ hold. In the *overhearing* approach to monitoring [10], the agent behaviours that bring about states of interest are inferred from the messages exchanged. This suffices when states are inherently described as ones in which messages have been sent and received (e.g.,$NormActivation$ in Example 1). However, it may not be possible, or indeed desirable, to describe and recognise such states exclusively on the basis of exchanged messages. Consider $NormCondition$ in Example 1, in which:

> F is obliged to deposit payment for G in S's bank account within 3 days of B's acceptance.

Fulfilment of this obligation can be recognised by observing for F's sending of a notification message to S, informing the latter that payment has been made. However, to motivate deployment of S in a normative system containing the norm $NormGoods$ (e.g., as a signatory to an electronic contract specifying $NormGoods$ as a contractual clause), S must be assured that enforcement mechanisms will be applied if F does not pay. If recognition of fulfilment is *solely* based on observing the above notification message, then this assurance equates with an assurance that the notification message is sent if and only if payment has in fact been made. Of course, any such assurance would be questionable given that, if F has not made the payment, F can avoid sanction by still sending the notification message, since this will suffice to indicate fulfilment of the obligation.

We thus propose that agents deployed in normative systems explicitly entrust particular observers to accurately relay observations to monitors[2], where these observations may be of messages exchanged *or* of properties that describe some state of interest. Thus, in the above example, agents F and S might explicitly entrust the bank itself to be an observer that reports to a monitoring agent that F has deposited the money. Unlike the case of the notification message sent by F, no gain accrues to the bank if the bank mis-reports.

Alternatively, agents F and S could also entrust an observer other than the bank, to accurately report on both a message sent from F to S notifying that payment has been made, *and* a message sent from the bank to S notifying the latter that the money has been deposited (although observation of messages sent from banks would raise information privacy issues that may prove difficult to resolve).

Finally, consider the case in which a supplier is obliged to place a purchased book on an internet site for downloading by a purchaser, within 14 days of receiving payment. In this case, the internet site can be ascribed trusted observer status, reporting to a monitor that the book has been placed.

3.2 The Monitoring Architecture

In this section we describe our architecture for monitoring the behaviours of agents deployed in normative systems. Trusted observers report to monitors on whether states of interests referenced by norms' components, do or do not hold. Monitors process these observations together with *augmented transition network* (*ATN*) [21] representations of norms. In Section 4 we describe in detail how these *ATN*s represent the states of interest.

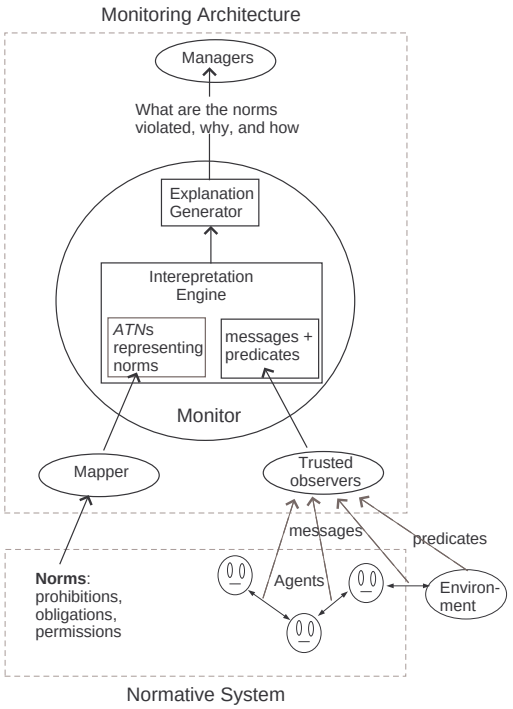

Figure 1: Monitoring architecture and its relationship to a normative system.

Figure 1 provides an overview of the monitoring architecture, in which the agents are treated as black boxes and their internal state transitions are invisible to the monitors. A mapper maps norms to their *ATN* representations, for input to the monitor (this input is provided off-line). At run time, monitors subscribe to all observers entrusted with reporting on the states of interest identified by a norm's components ($NormActivation$, $NormCondition$ and $NormExpiration$). The monitor can identify which observers to subscribe to, based on the *ATN* representation (as will be described in Section 4). Notice that there is nothing in the specification of the monitor that ties it to a particular normative system.

At run-time, observers notify monitors as to whether states of interest hold or not, by notifying monitors of: messages exchanged amongst agents, messages exchanged between agents and the environment, and predicates describing properties of the world (these properties may refer to actions having been performed). For example, in the use case implementation in Section 5, a state of interest in which an engine e has been repaired may be described in terms of a message sent from the repairer of the engine informing another party that the engine has been repaired, or an inferred predicate $engine\_repaired(e)$ from some theory describing the environment.

Notice that the observers are external to the normative system itself; their role is only to report on whether predicates hold or not, or on the sending and receipt of messages. They are not responsible for any kind of processing of this information. Thus, any environmental artifact can be assigned trusted observer status, including internet sites, human agents, banks, description logic reasoners, etc.

[2]For example by stipulating contractual agreements as to which observers are to be trusted

Monitors process observations together with the *ATN* representations of the norms, to determine when a norm is activated, fulfilled or violated, or has expired. Finally, the monitor informs manager agents of norms that have been violated, and of the agents responsible for violation. Manager agents then in turn impose sanctions on responsible agents.

Notice also that the choice of observers (and monitors) is application -specific, and agreed to by the agents whose behaviours are being observed, where such agreement constitutes a declaration of trust. The behaviours of observers (and monitors) may themselves be governed by normative clauses, and thus observed and monitored for deviation from their expected behaviour. This would reduce the potential for collusion (e.g., an agent dealing with eBay is more likely to trust a PayPal observer, even though PayPal is owned by eBay, if the behaviour of PayPal is itself normatively prescribed and sanctioned in case of violation).

4. REPRESENTATION AND PROCESSING OF NORMS FOR MONITORING

*ATN*s [21] are essentially directed labelled graphs that were originally proposed for parsing complex natural languages. In general, transitioning an arc from one *ATN* node to another is based on some processing of that arc's labels. A basic *ATN* representation of a norm $\mathcal{N}$ is a three node *ATN* ($\{S1, S2, S3\}$, $\mathcal{A}1, \mathcal{A}2$), where $\mathcal{A}1$ is the set of arcs connecting node $S1$ to node $S2$, and $\mathcal{A}2$ is the set of arcs connecting node $S2$ to node $S3$. Here, the transition from $S1$ to $S2$ corresponds to the activation of a norm, while the transition from $S2$ to $S3$ corresponds to the norm's fulfillment or violation. The arcs are labelled so that, based on reports received from observers, the monitor matches the reports with the arc labels that describe the states of interest specified by $\mathcal{N}$'s components, and then transitions the *ATN* from one node to the next, across the matched arc. If the *ATN* remains at node $S1$, then $\mathcal{N}$ is not activated. If the *ATN* is at node $S2$, then $\mathcal{N}$ is activated, and in the case of an obligation, if it remains at $S2$ after a given period of time identified by $NormCondition$, then $\mathcal{N}$ is violated (the deadline is exceeded). If the *ATN* is at node $S3$, then $\mathcal{N}$ is fulfilled if $\mathcal{N}$ is an obligation or permission, or violated if $\mathcal{N}$ is a prohibition (a prohibited state has been reached).

In what follows we describe in more detail the representation of norms as *ATN*s, and their processing by monitors.

4.1 Representing Norms as *ATN*s

A key requirement for norm representation and processing is to account for representation of complex behaviours and states of interest, possibly enacted and brought about jointly by multiple agents. For example, consider $NormGoods$ (Example 1) in which the state in which the obligation is fulfilled is described by a disjunction of behaviours, where the second disjunct:

B must accept the order within 7 days of receipt of notification **and** *F must deposit payment for G in S's bank account, within 3 days of B's acceptance*

describes a conjunction of behaviours by two distinct agents B and F, where these obliged behaviours must satisfy some temporal constraints. Recall also Section 3.1's suggested representation for detection of *F must deposit payment for G in S's bank account*, in terms of a notification message sent by F to S *and* a message from S's bank, to S, confirming receipt of the monies.

In order to be able to represent such complex descriptions of behaviours and states of interest, we assume representation of each norm component in disjunctive normal form. In Section 3 we described requirements for representing the states of interest described by each norm component, in terms of messages exchanged, and/or predicate descriptions. Such representations are obtained by a mapping from a disjunctive normal form formulation of each norm component. These representations are then used to label arcs in the *ATN* representation of a norm.

DEFINITION 1. [Mapping norm components to *ATN* labels]
For each norm component $\mathcal{N}Comp \in \{ NormActivation, NormCondition, NormExpiration \}$, let $\mathcal{N}Comp$ =

$$\alpha_1 \vee \ldots \vee \alpha_n,$$

where for $i = 1 \ldots n$, α_i is a conjunction:

$$\beta_1 \wedge \ldots \wedge \beta_m.$$

Then, for $j = 1 \ldots m$:

$$map(\beta_j) = \{(Ob_{\beta_j}, M_{\beta_j}, T_{\beta_j})\}, \text{ where:}$$

1. *M_{β_j} is a message or predicate description*
2. *T_{β_j} is a temporal expression to be evaluated in conjunction with the processing of M_{β_j}*
3. *Ob_{β_j} is a unique identifier for the observer responsible for observing and reporting M_{β_j}*

For any $\alpha_i = \beta_1 \wedge \ldots \wedge \beta_m$ we let

$$map\_conj(\alpha_i) = \bigcup_{j=1}^{m} map(\beta_j)$$

Notice that a predicate description may be of form $happened(Act)$ where Act is an action, or may describe the post-conditions of an action. The latter option may provide for more flexibility in terms of the actions executed to bring about the post-conditions describing the state of interest. While Ob_{β_j} uniquely identifies the observer entrusted to report the messages or the truth of predicates in M_{β_j}, in practice, multiple observer identifiers may refer to the same observer. Notice that the observer identifiers enable a monitor to identify which observers to subscribe to when processing *ATN*s.

EXAMPLE 2. *For $NormActivation$ in Example 1, we obtain* $\{(Ob_{S\_notify}, (send(S, B, in\_stock(G)), T1)^3, T1)\}$

and for the first disjunct in $NormCondition$:

$\{$
$(Ob_{B\_cancel}, (send(B, S, cancel(G)), T2), T2 \preceq (T1 + 7))$ $\}$

and for the second disjunct in $NormCondition$:

$\{$
$(Ob_{B\_accept}, (send(B, S, accept(G)), T3), T3 \preceq (T1 + 7)),$
$(Ob_{S\_bank}, (deposited(F, S, Gcost), T4), T4 \preceq (T3 + 3))$
$\}$

We now define representation of a norm as an *ATN*:

DEFINITION 2. [Defining *ATN* representations of norms]
Let $\mathcal{N}$ = ($NormType$,$NormActivation$,$NormCondition$, $NormExpiration$,$NormTarget$), where:

- $NormActivation = \delta_1 \vee \ldots \vee \delta_m$
- $NormCondition = \epsilon_1 \vee \ldots \vee \epsilon_n$

The defined ATN *is a three node labelled graph represented as a tuple ($\{S1, S2, S3\}$,$\mathcal{A}1, \mathcal{A}2$) where:*

- *$\mathcal{A}1$ is set of arcs $\{(S1, S2)_1, \ldots, (S1, S2)_m\}$ such that for $i = 1 \ldots m$, $map\_conj(\delta_i)$ labels $(S1, S2)_i$*
- *$\mathcal{A}2$ is set of arcs $\{(S2, S3)_1, \ldots, (S2, S3)_n\}$ such that for $i = 1 \ldots n$, $map\_conj(\epsilon_i)$ labels $(S2, S3)_i$*

Figure 2a illustrates an *ATN* representation of a generic norm $\mathcal{N}$, and Figure 2b shows the *ATN* representation of $NormGoods$.

[3]We assume FIPA communication standards, and represent message descriptions as tuples consisting of the message itself and its time stamp

[h]

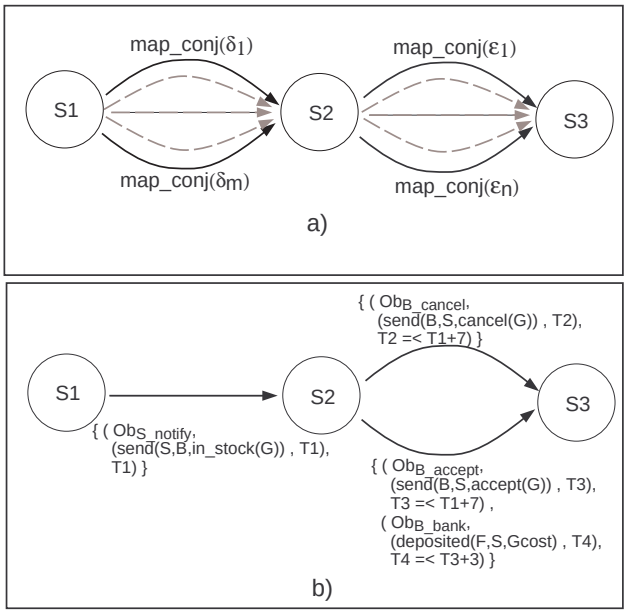

Figure 2: *ATN* representations of norms

4.2 Interpretation of *ATN*s by Monitors

This section describes how monitors process reports received from observers, together with *ATN* representations of norms, in order to determine the status of norms.

Consider a norm $\mathcal{N}$ represented as the *ATN* ($\{S1, S2, S3\}$, $\mathcal{A}1$, $\mathcal{A}2$). If at least one arc in $\mathcal{A}1$ or $\mathcal{A}2$ is *satisfied* — by which we mean that observers report that the messages labelling the arc have been sent, and / or the predicate descriptions hold, and the temporal expressions labelling the arc are evaluated by the monitor to be true — then the monitor transitions the arc to the successor node (where $S2$ and $S3$ are the successors of $S1$ and $S2$ respectively).

If an arc δ_i in $\mathcal{A}1$ is transitioned from $S1$ to $S2$, the monitor can communicate to a manager that $\mathcal{N}$ is activated, where the reasons for the activation are the messages and predicates labelling δ_i. If an arc ϵ_i in $\mathcal{A}2$ is transitioned from $S2$ to $S3$, and $\mathcal{N}$ is a permission or obligation, then the monitor can communicate to a manager that $\mathcal{N}$ is fulfilled, and if $\mathcal{N}$ is a prohibition, then the monitor can communicate to a manager that $\mathcal{N}$ is violated. In either case, the reasons for the fulfillment or violation — the messages and predicates labelling ϵ_i — can also be communicated to the manager.

Finally, $NormExpiration$ represents conditions under which, if an *ATN* is in node $S1$, the norm is not activated (irrespective of whether an arc in $\mathcal{A}1$ is satisfied), and if the *ATN* is in state $S2$, then the issue of the norm's fulfillment / violation no longer arises.

4.2.1 Time windows for detecting violation of obligations

We have thus far not considered the case in which an obligation is violated. We aim at representation and monitoring of obligations whose violation can be determined with respect to some temporal condition holding. For example, in order that one can determine whether an obligation to bring about some state (an achievement obligation) is violated, reference is required to some point of time before which the state must be brought about. Some maintenance obligations also explicitly reference a time period during which some state of affairs must be maintained.

We thus define the notion of a *time window*, based on the temporal expressions labelling the arcs in $\mathcal{A}2$. Consider a modified version of Example 1 in which we simply have the obligation β_1 = 'B is obliged to cancel within 7 days of receipt of notification'. This obligation can only be violated *after* 7 days after receipt of notification; 7 days is the time window for this obligation. Consider also a modified version of Example 1 in which $\beta_2 \wedge \beta_3$ = 'B is obliged to accept within 4 days of notification *and* F pays within 6 days of notification'. This obligation can only be violated after 4 days after receipt of notification. Notice that if it is day 5, and B has not accepted, then the obligation is violated. Now let us put these normative conditions together in a disjunct: $\beta_1 \vee (\beta_2 \wedge \beta_3)$. The time window in which the monitor must check for violation is now the maximum time window of all the disjuncts, i.e., 7 days. To illustrate, consider the following scenarios:

1. Suppose it is day 6 and neither disjunct is satisfied. The obligation is not violated since B still has a day left in which to cancel.
2. Suppose it is day 8 and B has accepted on day 5 after the notification and F has paid on day 5 after the notification. The obligation is violated since neither disjunct is satisfied.
3. Suppose it is day 7 and B has accepted on day 4 after the notification and F has not yet paid. The obligation is not violated since it can still be fulfilled by B by cancelling on day 7[4].

In general then, we assume a defined function $time\_window$ that returns a temporal expression (e.g. '7 days after receipt of notification') which, if evaluated to true, and if none of the $NormCondition$'s arcs (in $\mathcal{A}2$) are satisfied, then if the norm is an obligation, the norm is said to be violated.

DEFINITION 3. [Signatures of functions defining time windows] *Let* $\mathcal{A}2 = \{(S2, S3)_1, \ldots, (S2, S3)_n\}$. *Then:*

- $time\_w\_d : (S2, S3)_i \mapsto TWD$ *where* TWD *is a temporal expression.*
- $time\_window : \{ time\_w\_d((S2, S3)_1), \ldots, time\_w\_d((S2, S3)_n)\} \mapsto TW$ *where* TW *is a temporal expression*

From the preceding example, one can see that an obvious way to define these functions is by letting $time\_w\_d$ return some minimal temporal value for temporal expressions in the conjuncts labelling a single arc, and $time\_window$ returns the maximum amongst these values. In what follows, we will as an abuse of notation write $time\_window(\mathcal{A}2)$ to denote the temporal expression returned by the function $time\_window$ in the above definition.

4.2.2 Formally defining interpretation of ATNs

We now define the above concepts formally, in which we assume a monitor Mon interpreting an *ATN* $\mathcal{X}_\mathcal{N}$ = ($\{S1, S2, S3\}$,$\mathcal{A}1$, $\mathcal{A}2$) for the norm $\mathcal{N}$ = ($Norm\ Type$,$Norm\ Activation$, $NormCondition$,$Norm\ Expiration$, $NormTarget$)

DEFINITION 4. [Satisfaction of arcs]
Let a *be an arc in* $\mathcal{A}1$ *or* $\mathcal{A}2$*, where* a *is labelled by:*

$$\{ (Ob_{\beta_1}, M_{\beta_1}, T_{\beta_1}), \ldots, (Ob_{\beta_n}, M_{\beta_n}, T_{\beta_n})\}$$

We say that a *is* satisfied*, if for* $i = 1 \ldots n$, Mon *receives* M_{β_i} *from* Ob_{β_i}*, and* T_{β_i} *evaluates to true.*

[4]This illustrates that one should additionally prohibit an agent from both accepting *and* cancelling.

DEFINITION 5. [Interpreting $NormExpiration$]
Let $NormExpiration = \gamma_1 \vee \ldots \vee \gamma_l$.

- *We say that a monitor* Mon *evaluates* $NormExpiration$ *as not holding if, for* $j = 1 \ldots l$, γ_j *does not not hold.*
- *We say that* $\gamma_j = (\beta_1 \wedge \ldots \wedge \beta_m)$ *does not not hold if:*
  - *for at least one* i, *where* map$(\beta_i) = (Ob_{\beta_i}, M_{\beta_i}, T_{\beta_i})$, *either* Mon *does not receive* M_{β_i} *from* Ob_{β_i}, *or* T_{β_i} *does not evaluate to true.*

DEFINITION 6. [Transitioning ATN to $S2$]
Let $\mathcal{N}$ *not be activated, fulfilled, or violated (*$\mathcal{X}_\mathcal{N}$ *is in node* $S1$*). Suppose monitor* Mon *evaluates* $NormExpiration$ *as not holding. Then, if at least one arc* a *in* $\mathcal{A}1$ *is* satisfied, $\mathcal{X}_\mathcal{N}$ *transitions to* $S2$ *and* $\mathcal{N}$ *is said to be activated.*

DEFINITION 7. [Transitioning ATN to $S3$]
Let $\mathcal{N}$ *be activated (*$\mathcal{X}_\mathcal{N}$ *is in node* $S2$*). Suppose monitor* Mon *evaluates* $NormExpiration$ *as not holding.*

If at least one arc a *in* $\mathcal{A}2$ *is* satisfied, *then* $\mathcal{X}_\mathcal{N}$ *transitions to* $S3$, *and:*

- *If* $\mathcal{N}$ *is an obligation or permission, then* $\mathcal{N}$ *is said to be fulfilled.*
- *If* $\mathcal{N}$ *is a prohibition then* $\mathcal{N}$ *is said to be violated.*

DEFINITION 8. [Violation of obligation in $S2$]
Let the obligation $\mathcal{N}$ *be activated (*$\mathcal{X}_\mathcal{N}$ *is in node* $S2$*). Suppose* Mon *evaluates* $NormExpiration$ *as not holding.*

If $time\_window(\mathcal{A}2)$ *evaluates to true, and no arc in* $\mathcal{A}2$ *is satisfied, then* $\mathcal{X}_\mathcal{N}$ *remains in node* $S2$, *and the obligation* $\mathcal{N}$ *is said to be violated.*

We have thus far illustrated monitoring with respect to the obligation $NormGoods$ in Example 1. In the next section we describe an implementation of a monitor, and its validation under a real-world scenario that involves agents governed by permissions, prohibitions and obligations in the aerospace domain.

5. IMPLEMENTATION AND VALIDATION

5.1 Monitor Implementation

This section describes an implementation of a monitor that receives messages from observers, and processes them so as to transition the *ATN* representations of the norms being monitored. At its core, our monitor contains a message store that is updated by received messages. When an arc is satisfied (see Definition 4) with respect to the contents of a message store, the monitor transitions the *ATN*. We consider two types of *ATN*: *abstract* and *instantiated*. *ATN*s in state $S1$ are said to be abstract because their arcs are labelled by expressions whose variables will be instantiated by concrete situations in which the norm comes into force (is activated). Hence, given an abstract *ATN*, when an arc a in $\mathcal{A}1$ is satisfied, the resulting grounding of the variables in the M_βs labelling a is propagated to the variables in expressions labelling arcs in $\mathcal{A}2$, thus creating an instantiated instance I of the abstract *ATN*, where I is then transitioned to $S2$ (corresponding to activation of the norm).

For example, the norm in Example 1 has a variable denoting a generic type of good that, once accepted, needs to be paid for. In this case, when the monitor receives a message denoting that a supplier *Susan* has the good *galoshes* in stock for buyer *Bernard*, the abstract *ATN* of Figure 2b is instantiated and its variables *S*, *G* and *B* are bound to *Susan*, *galoshes* and *Bernard* respectively. From this point on, only observer messages whose contents satisfy these variable groundings can cause the instantiated *ATN* to be transitioned to state $S3$.

Finally, when norms are fulfilled or violated, the monitor generates notifications to the manager to take appropriate action. We illustrate this operation in Figure 3, which shows the flow of messages from the observers to the monitor's message queue, its processing and subsequent notifications to the manager.

Algorithm 1 Monitor control loop

```
1: while Monitor is active do
2:   while Q_msg is not empty do
3:     Retrieve Msg from head of Q_msg
4:     Add Msg to M_St{First, deal with messages}
5:     for all Abstract norm ATN A in X_Abs do
6:       for all Arcs a in A1_A do
7:         if satisfied(M_St,arc label a) then
8:           create a norm ATN instance I of A
9:           add I to X_Inst
10:          move I to state S2
11:        end if
12:      end for
13:    end for
14:    for all Instantiated norm ATN I in X_Inst do
15:      for all Arcs a in A2_I do
16:        if satisfied(M_St,arc label a) then
17:          remove I from X_Inst
18:          move I to state S3
19:          if Norm I is an obligation or permission then
20:            notify manager of fulfilment
21:          else if Norm I is a prohibition then
22:            notify manager of violation
23:          end if
24:        end if
25:      end for
26:    end for
27:  end while{Now deal with time windows}
28:  for all Instantiated norm ATN I in X_Inst do
29:    if time_window(A2_I) and I is an obligation then
30:      notify manager of violation
31:    end if
32:  end for
33: end while
```

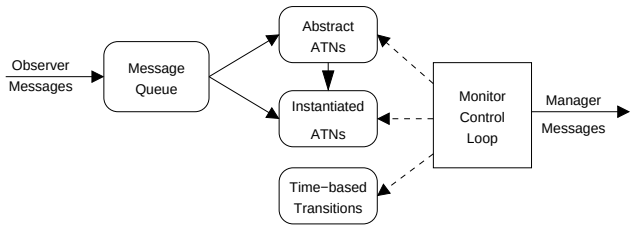

Figure 3: Overview of the monitor control loop.

This process is more precisely illustrated in Algorithm 1, which describes the control loop used in our monitor.

Its initial loop (lines 2 to 27) handles messages received from the observers by initially trying to satisfy outgoing arcs from $S1$ in the abstract *ATN*s (line 6). If successful for at least one arc (line 7), then an instantiated *ATN* is created, added to the set of instantiated *ATN*s and transitioned to $S2$ (lines 8 to 10). Satisfaction of out-

going arcs from $S2$ (i.e., arcs in $\mathcal{A}2$) in each instantiated *ATN* is then checked (line 15), and if successful, the instantiated *ATN* is removed from the set of instantiated *ATN*s and transitioned to $S3$ (lines 17 to 18). When this transition occurs and the norm is an obligation or a permission, it denotes that the norm has been fulfilled, else if the norm is a prohibition the norm has been violated, and the manager is notified accordingly (lines 19 to 22).

Finally, our algorithm needs to deal with the case where a time window has elapsed for obligations. This occurs in the loop of lines 28 through 32. If an instantiated obligation *ATN* is still in $\mathcal{X}_{Inst}$ (none of its arcs in $\mathcal{A}2$ are satisfied), and $time\_window$ ($\mathcal{A}2$) evaluates to true, then the monitor notifies the manager that the obligation has been violated.

5.2 Use Case

In order to validate our approach to monitoring, we implemented and deployed a monitor in a prototype multi-agent system in which agents exchange messages that correspond to obliged, prohibited and permitted behaviours encoded in an electronic contract. Specifically, recent work on electronic representations and software tools for contracts [18] have highlighted a number of case studies [7]. A prototype for an aerospace logistics case study [16] implements aerospace agents — airline operators (*AO*s), engine manufacturers (*EM*s), and service sites (*SS*s) — whose behaviours are required to comply with (amongst others) norms governing the repair of engines and sourcing of parts for these repairs. In particular, it is commonplace for *EM*s, located at airports, to be under obligation to have operational engines available for the planes of a client *AO*. Furthermore, *AO*s may dictate permissions and prohibitions on the sourcing of parts for their engines. These norms are then inherited in contracts between *EM*s and service sites responsible for the actual servicing and repair of engines. For instance, in order for a given *EM Boing* to fulfill its obligations and provenance restrictions for a given *AO*, *Boing*'s contract C with a service site *Heathhedge* stipulates the following norms:

1. **Ob1**: $Heathhedge$ is obliged to repair engines for $Boing$ within 7 days of receipt of an order for repair.
2. **Per1** and **Per2**: $Heathhedge$ is permitted to source parts for engines for $Boing$, from part manufacturers 1 and 2 ($pm1$ and $pm2$).
3. **Pro3**: $Heathhedge$ is prohibited from sourcing parts for engines for $Boing$, from part manufacturer 3 ($pm3$).

The prototype implementation deploys *AgentSpeak(L)* agents [19] representing each of the contract parties $Boing$ and $Heathhedge$, as well as the part manufacturers $pm1$, $pm2$ and $pm3$. We also deploy observers, a manager, and a monitor. The observers are responsible for relaying to the monitor, messages exchanged between $Boing$ and $Heathhedge$, and between $Heathhedge$ and the part manufacturers. In what follows we describe two scenarios in which messages are observed and relayed to a *monitor* who then processes these together with *ATN*s representing the above norms, and reports on their status. For both scenarios it is assumed that there is a prior message from *Boing* to $Heathhedge$ ordering repair of an engine. This message results in the *ATN*s for each of **Ob1**, **Per1**, **Per2** and **Pro3** being transitioned to their activation state $S2$.

Scenario 1:

1 $Heathedge$ orders a part for the engine from *pm1*.

2 $pm1$ informs $Heathedge$ that the delivery time for the part is 3 days.

3 The delivery time is acceptable for $Heathedge$ since it will allow $Heathedge$ to repair the engine within 7 days, and so $Heathedge$ orders the part from $pm1$. This order message results in the ATN for **Per1** transitioning to $S3$, thus indicating fulfilment of the permission.

4 However, $Heathedge$ is notified by $pm1$ that the part has been sent on the 4th day after the part order from $Heathedge$ (because of unavoidable delays).

5 Because of the delay in receipt of the part, $Heathhedge$ cannot repair the engine repair within 7 days and so no message informing completion of repair is sent from $Heathhedge$ to $Boing$ within the 7 day time window. Hence the ATN for **Ob1** is in state $S2$ on day 8, and the monitor informs the manager that $Heathhedge$ has violated its obligation.

In Scenario 2, $pm1$ and $pm2$ inform $Heathedge$ of delivery times that are not acceptable. $Heathedge$ is then faced with a choice of violating its obligation, or ordering from the prohibited $pm3$ who can provide the part in an acceptable time. $Heathhedge$ chooses the latter option. Observation of the order message sent from $Heathedge$ to $pm3$ results in transitioning the *ATN* for **Pro3** to $S3$, and the monitor informs the manager of violation of this prohibition. However, $Heathhedge$ informs $Boing$ of completion of repair on the sixth day after receipt of the repair order. This results in the *ATN* for **Ob1** transitioning to $S3$. Hence, the obligation is reported by the monitor as fulfilled.

6. CONCLUSIONS

In this paper we have described a monitoring architecture in which trusted observers report to monitors on states of interest relevant to the activation, fulfillment, violation and expiration status of norms. This provides some measure of assurance that sanctions will be applied as and when appropriate, and thus enhances prospects for deployment of agents in normative systems. We have described how individual norms are represented as *ATN*s, that are processed by monitors together with observations relayed by observers, in order to determine the status of norms. The *ATN*s represent complex behaviours and states of interest, and only behaviours specified by the norms are represented. Hence, a given *ATN* can represent the same norm specified in any one of a number of multi-agent systems. Furthermore, *ATN* representations of norms are independent of each other, allowing run time addition and removal of norms to the monitored system. Taken together, these features enhance the generality of the framework, and to substantiate this claim, we have described a proof of concept implementation of a monitor agent monitoring agents whose behaviours are governed by normative clauses in an electronic contract.

The framework described in this paper extends a recent preliminary framework [4] in that it allows for monitoring of complex jointly realised behaviours, deploys trusted observers, and implements and validates processing *ATN* representations of norms. Other work related to ours includes monitoring of contracts in a Web Services context [11, 17], where the focus is on quality of service metrics rather than on the behaviours of agents. Other researchers adopt an overhearing approach to monitoring in organisational contexts [13, 10]. However, these works adopt overhearing in order to infer the mental states of the agents, where these states are domain-dependent and private to the agents. By contrast, we adopt overhearing of messages exchanged *and* predicates, for the different purpose of evaluating the status of norms.

We conclude with a discussion of three areas of future work to be pursued. As mentioned in Section 2, further work needs to address representation and monitoring of maintenance obligations that do not explicitly refer to temporal conditions. For example, consider an obligation to always drive on the left. In practice, such a norm

will be activated at a time $T1$ when an individual starts to drive, and terminates at time $T2$ when she ceases to drive. One might then specify an explicit temporal window: $\forall T, T1 \leq T \leq T2$. This would require extending the processing of an *ATN* to transition back and forth from $S2$ to $S3$. At each time point T, if she is driving on the left, the *ATN* transitions to the fulfillment state $S3$, and then immediately transitions back to $S2$, so that if at the next time point she is not driving on the left, then she can be detected as having violated the obligation. An alternative would be to create a new instantiation of the maintenance obligation at each time point T. In this view, the fulfillment of an instance of the norm at T results in its expiration, and so a new instance is created at $T + 1$.

Providing explanations of norm violations is important if managers are to appropriately assign responsibility and apply sanctions. Explanations can also help to evolve the normative specification of a system in order to prevent future violations. Thus far, monitors provide limited explanation of violation and fulfilment, in terms of the labels of the arcs transitioned. Future work will investigate generation of more comprehensive explanations. This may require reference to representations of work-flow separate from that implicitly specified by norms. This information may in turn be gleaned from observers. For example, in the second scenario in the use case, access to all messages exchanged between $Heathedge$ and the part manufacturers can be used to explain that in order to fulfill the repair obligation, the prohibition on sourcing parts from $pm3$ had to be violated, since $pm1$ and $pm2$'s delivery times would not have allowed the obligation to be fulfilled.

Finally, we note that the focus of this paper has been on *corrective* monitoring, whereby critical states are monitored for violation of norms. *Predictive* monitoring requires representation and recognition of danger states, which are associated with agent behaviours that suggest that a norm may be in danger of violation. Future work will address how such states may be identified empirically; for example by observing and analysing violation of norms at run-time and the intermediate states that are reached prior to violation. These intermediate states can then be explicitly (as extra nodes) in the *ATN* representation of norms, so that during future run-time executions, observation of messages indicating transition to these states may signal preemptive action to avoid violation.

Acknowledgements: The research described in this paper is partly supported by the European Commission Framework 6 funded project CONTRACT (INFSO-IST-034418). The opinions expressed herein are those of the named authors only and should not be taken as necessarily representative of the opinion of the European Commission or CONTRACT project partners.

7. REFERENCES

[1] R. Conte, R. Falcone, and G. Sartor. Agents and norms: How to fill the gap? *Artificial Intelligence and Law*, 7:1–5, 1999.

[2] M. Dastani, D. Grossi, J.-J. C. Meyer, and N. Tinnemeier. Normative multi-agent programs and their logics. In *Proc. Workshop on Knowledge Representation for Agents and Multi-Agent Systems (KRAMAS'08)*, pages 236–243, 2008.

[3] M. Esteva, B. Rosell, J. A. Rodríguez-aguilar, and J. L. Arcos. Ameli: An agent-based middleware for electronic institutions. In *3rd Int. Joint Conference on Autonomous Agents and Multi-agent Systems*, pages 236–243, 2004.

[4] N. Faci, S. Modgil, N. Oren, F. Meneguzzi, S. Miles, and M. Luck. Towards a monitoring framework for agent-based contract systems. In *12th Int. Workshop on Cooperative Information Agents (CIA 2008)*, pages 292–305, 2008.

[5] A. D. H. Farrell, M. Sergot, M. Salle, and C. Bartolini. Using the event calculus for tracking the normative state of contracts. *International Journal of Cooperative Information Systems*, 4(2–3):99–129, 2005.

[6] D. Grossi. *Designing Invisible Handcufffs*. PhD thesis, Utrecht University, SIKS, 2007.

[7] M. Jakob, M. Pchouek, J. Chabera, S. Miles, M. Luck, N. Oren, M. Kollingbaum, C. Holt, J. Vazquez, P. Storms, and M. Dehn. Case studies for contract-based systems. In *Proc. 7th Int. Joint Conference on Autonomous Agents and Multiagent Systems*, pages 55–62, 2008.

[8] N. R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence*, 75(2):195–240, 1995.

[9] A. J. I. Jones and M. Sergot. On the characterisation of law and computer systems: The normative systems perspective. In *Deontic Logic in Computer Science: Normative System Specification*, pages 275–307. John Wiley and Sons, 1993.

[10] G. Kaminka, D. Pynadah, and M. Tambe. Monitoring teams by overhearing: A multi-agent plan-recognition approach. *Journal of Artificial Intelligence Research*, 17:83–135, 2002.

[11] A. Keller and H. Ludwig. The WSLA framework: Specifying and monitoring service level agreements for web services. *Journal of Network Systems Management*, 11(1):57–81, 2003.

[12] M. Kollingbaum. *Norm-governed Practical Reasoning Agents*. PhD thesis, University of Aberdeen, 2005.

[13] F. Legras and C. Tessier. Lotto: group formation by overhearing in large teams. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 425–432, 2003.

[14] F. L. Y. Lopez, M. Luck, and M. d'Inverno. A normative framework for agent-based systems. *Computational and Mathematical Organization Theory*, 12(2-3):227–250, 2006.

[15] H. Mazouzi, A. E. F. Seghrouchni, and S. Haddad. Open protocol design for complex interactions in multi-agent systems. In *1st Int. Joint Conference on Autonomous Agents and Multiagent Systems*, pages 517–526, 2002.

[16] F. R. Meneguzzi, S. Miles, M. Luck, C. Holt, M. Smith, N. Oren, N. Faci, M. Kollingbaum, and S. Modgil. Electronic contracting in aircraft aftercare: A case study. In *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems, Industry and Applications Track*, 2008.

[17] Z. Milosevic, S. Gibson, P. Linington, J. Cole, and S. Kulkarni. On design and implementation of a contract monitoring facility. In *Proceedings of the First International Workshop on Electronic Contracting*, page 10. IEEE, 2004.

[18] N. Oren, S. Panagiotidi, J. Vazquez-Salceda, S. Modgil, M. Luck, and S. Miles. Towards a formalisation of electronic contracting environments. In *To appear in Proc. Coordination, Organization, Institutions and Norms in Agent Systems (COIN 2008)*, 2008.

[19] A. S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In *Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*. Springer, 1996.

[20] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[21] W. A. Woods. Transition network grammars for natural language analysis. *Communications of the ACM*, 13(10):591–606, 1970.